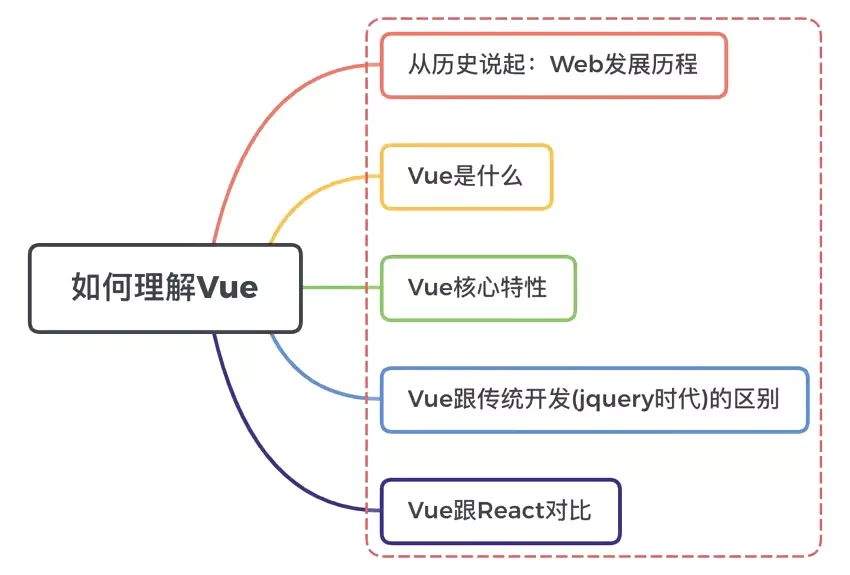
# 面试官：有使用过vue吗？说说你对vue的理解



## 一、从历史说起

Web是World Wide Web的简称，中文译为万维网我们可以将它规划成如下的几个时代来进行理解

* 石器时代
* 文明时代
* 工业革命时代
* 百花齐放时代

### 石器时代

石器时代指的就是我们的静态网页，可以欣赏一下1997的Apple官网



最早的网页是没有数据库的，可以理解成就是一张可以在网络上浏览的报纸，直到CGI技术的出现通过 CGI Perl 运行一小段代码与数据库或文件系统进行交互，如当时的Google（1998年）



### 文明时代

ASP，JSP大家应该都不会太陌生，最早出现于 2005 年左右，先后出现了微软的 ASP 和 Java Server Pages [JSP] 等技术,取代了 CGI ，增强了 WEB 与服务端的交互的安全性，类似于下面这样，其实就是Java + HTML

<%@ page language="java" contentType="text/html; charset=utf-8"     pageEncoding="utf-8"%> <!DOCTYPE html> <html> <head>   <meta charset="utf-8">   <title>JSP demo</title> </head> <body>   <img src="http://localhost:8080/web05\_session/1.jpg" width=200 height=100 /> </body> </html>

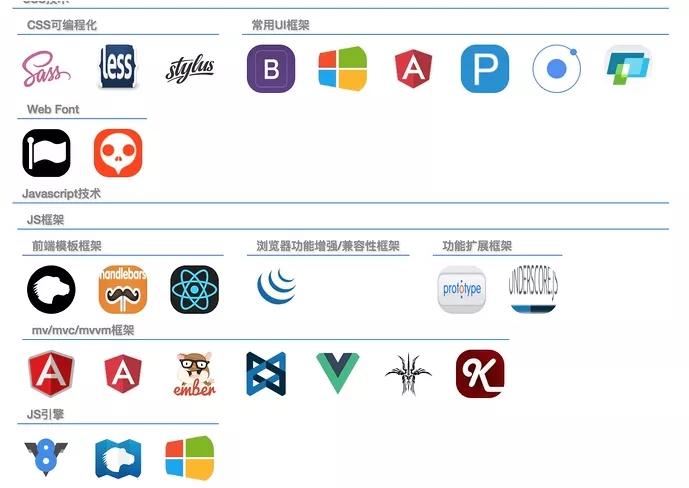
JSP有一个很大的缺点，就是不太灵活，因为JSP是在服务器端执行的，通常返回该客户端的就是一个HTML文本。我们每次的请求：获取的数据、内容的加载，都是服务器为我们返回渲染完成之后的 DOM，这也就使得我们开发网站的灵活度大打折扣在这种情况下，同年：Ajax火了(小细节，这里为什么说火了，因为 Ajax 技术并不是 2005 年出现的，他的雏形是 1999 年)，现在看来很常见的技术手段，在当时可是珍贵无比

### 工业革命时代

到这里大家就更熟悉了，移动设备的普及，Jquery的出现，以及SPA（Single Page Application 单页面应用）的雏形，Backbone EmberJS AngularJS 这样一批前端框架随之出现，但当时SPA的路不好走，例如SEO问题，SPA 过多的页面、复杂场景下 View 的绑定等，都没有很好的处理经过这几年的飞速发展，节约了开发人员大量的精力、降低了开发者和开发过程的门槛，极大提升了开发效率和迭代速度，我们可以称之其为工业时代

### 百花齐放时代

这里没有文字，放一张图感受一下



PS：这里为什么要说这么多Web的历史，我们可以看到Web技术的变化之大与快，每一种新的技术出现都是一些特定场景的解决方案，那我们今天的主角Vue又是为了解决什么呢？我们接着往下看

## 二、vue是什么

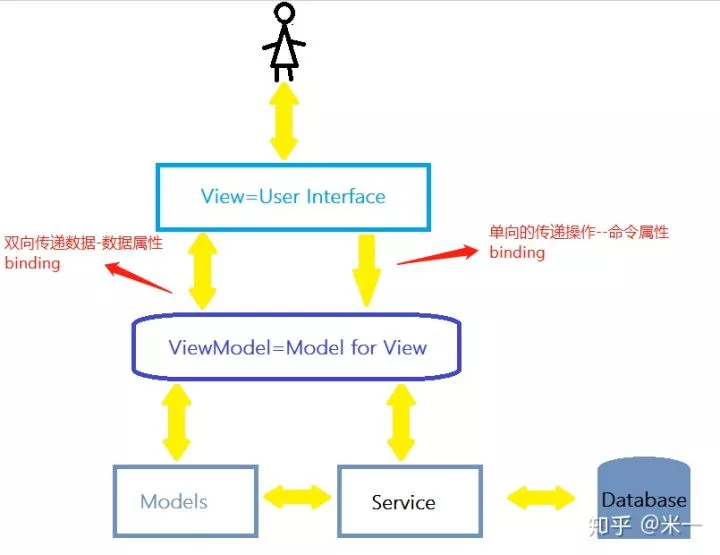
Vue.js（/vjuː/，或简称为Vue）是一个用于创建用户界面的开源JavaScript框架，也是一个创建单页应用的Web应用框架。2016年一项针对JavaScript的调查表明，Vue有着89%的开发者满意度。在GitHub上，该项目平均每天能收获95颗星，为Github有史以来星标数第3多的项目同时也是一款流行的JavaScript前端框架，旨在更好地组织与简化Web开发。Vue所关注的核心是MVC模式中的视图层，同时，它也能方便地获取数据更新，并通过组件内部特定的方法实现视图与模型的交互PS: Vue作者尤雨溪是在为AngularJS工作之后开发出了这一框架。他声称自己的思路是提取Angular中为自己所喜欢的部分，构建出一款相当轻量的框架最早发布于2014年2月

## 三、Vue核心特性

### 数据驱动（MVVM)

MVVM表示的是 Model-View-ViewModel

* Model：模型层，负责处理业务逻辑以及和服务器端进行交互
* View：视图层：负责将数据模型转化为UI展示出来，可以简单的理解为HTML页面
* ViewModel：视图模型层，用来连接Model和View，是Model和View之间的通信桥梁

这时候需要一张直观的关系图，如下 

### 组件化

1.什么是组件化一句话来说就是把图形、非图形的各种逻辑均抽象为一个统一的概念（组件）来实现开发的模式，在Vue中每一个.vue文件都可以视为一个组件2.组件化的优势

* 降低整个系统的耦合度，在保持接口不变的情况下，我们可以替换不同的组件快速完成需求，例如输入框，可以替换为日历、时间、范围等组件作具体的实现
* 调试方便，由于整个系统是通过组件组合起来的，在出现问题的时候，可以用排除法直接移除组件，或者根据报错的组件快速定位问题，之所以能够快速定位，是因为每个组件之间低耦合，职责单一，所以逻辑会比分析整个系统要简单
* 提高可维护性，由于每个组件的职责单一，并且组件在系统中是被复用的，所以对代码进行优化可获得系统的整体升级

### 指令系统

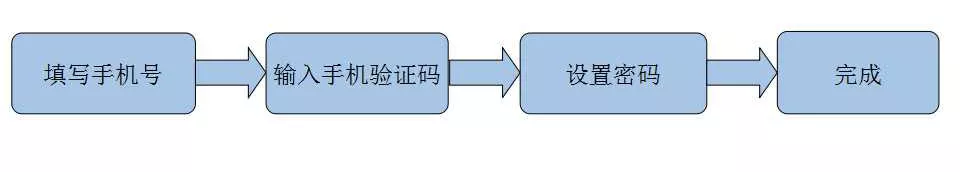
解释：指令 (Directives) 是带有 v- 前缀的特殊属性作用：当表达式的值改变时，将其产生的连带影响，响应式地作用于 DOM

* 常用的指令
* 条件渲染指令 v-if
* 列表渲染指令v-for
* 属性绑定指令v-bind
* 事件绑定指令v-on
* 双向数据绑定指令v-model

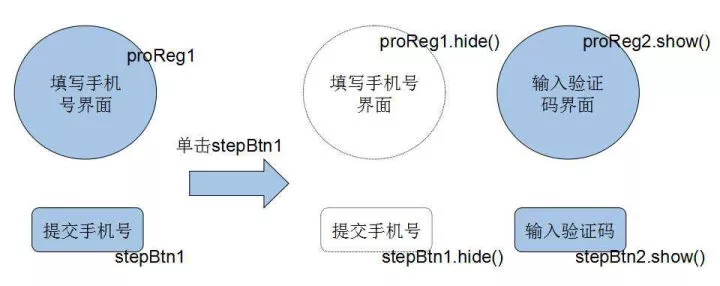
没有指令之前我们是怎么做的？是不是先要获取到DOM然后在....干点啥

## 四、Vue跟传统开发的区别

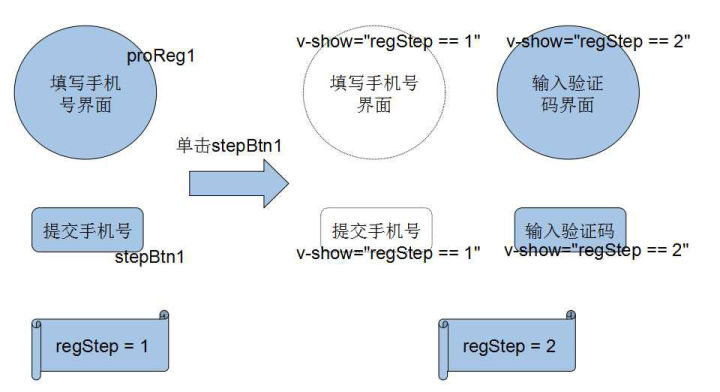
没有落地使用场景的革命不是好革命，就以一个高频的应用场景来示意吧注册账号这个需求大家应该很熟悉了，如下



用jquery来实现大概的思路就是选择流程dom对象，点击按钮隐藏当前活动流程dom对象，显示下一流程dom对象如下图(代码就不上了，上了就篇文章就没了..)



用vue来实现，我们知道vue基本不操作dom节点， 双向绑定使dom节点跟视图绑定后，通过修改变量的值控制dom节点的各类属性。所以其实现思路为：视图层使用一变量控制dom节点显示与否，点击按钮则改变该变量，如下图



总结就是：

* Vue所有的界面事件，都是只去操作数据的，Jquery操作DOM
* Vue所有界面的变动，都是根据数据自动绑定出来的，Jquery操作DOM

## 五、Vue和React对比

这里就做几个简单的类比吧，当然没有好坏之分，只是使用场景不同

### 相同点

* 都有组件化思想
* 都支持服务器端渲染
* 都有Virtual DOM（虚拟dom）
* 数据驱动视图
* 都有支持native的方案：Vue的weex、React的React native
* 都有自己的构建工具：Vue的vue-cli、React的Create React App

### 区别

* 数据流向的不同。react从诞生开始就推崇单向数据流，而Vue是双向数据流
* 数据变化的实现原理不同。react使用的是不可变数据，而Vue使用的是可变的数据
* 组件化通信的不同。react中我们通过使用回调函数来进行通信的，而Vue中子组件向父组件传递消息有两种方式：事件和回调函数
* diff算法不同。react主要使用diff队列保存需要更新哪些DOM，得到patch树，再统一操作批量更新DOM。Vue 使用双向指针，边对比，边更新DOM

## 参考文献

* https://segmentfault.com/a/1190000016269636
* https://zh.wikipedia.org/zh-cn/Vue.js
* https://zhuanlan.zhihu.com/p/20197803
* https://zhuanlan.zhihu.com/p/38296857

